Test Framework – 1

# Introduction

As per current scenario, whenever there is a data loading happens developers have to manually perform some initial data sanity check before business starts using it.

The test framework aims to build an automated system which will perform all the data basic sanity check’s after the data got loaded into target tables.

This framework will be performing below checks:

1. Null Check
2. Duplicate Check
3. Absolute Check

Output of the test framework will be written logs which thedeveloper/ tester can check once the process is completed.

## Null Check:

Check whether a particular column is null or not.

This will iteratively check on all the columns if it is null or not.

This test will raise an error if a particular column is null i.e. all the rows in the column contains null.

If some records of the column is null, it is not going to raise any error.

## Duplicate Check:

Sometimes, due to some wrong extractions or some failure, duplicate records get’s inserted into the table. This may cause a severe issue in production. Duplicate records need to be identified and resolved.

Here, Framework will check for the duplicate records based on the unique key provided, and push the data into a desired location.

For this, developer has to provide the unique key.

## Absolute Duplicate Check

If data gets loaded twice due to some issues, the developer has to manually sit and identify all the records which are loaded twice.

Here, framework will check if the table has any absolute duplicate record and write it into specified location.

# Configuration File

{

"test":

{

"absoluteDuplicateCheck":true/false,

"absoluteDuplicateCheckFileName":"/user/tsipl1305/testFramework/absduplicates",

"duplicateCheck":false,

"duplicateCheckColumns":"<list of columns>",

"duplicateCheckFileName":"/user/tsipl1305/testFramework/duplicates",

"nullCheck":true

},

"input":

{

"isDirectHive":true,

"driverName":"",

"databaseUrl":"",

"databaseUserName":"",

"databasePassword":"",

"query":"select \* from <table\_name>",

"isFile":false,

"fileName":"",

"fileWithSchema":false

},

"log":

{

"fileName":"/home/<username>/sanity\_check.log"

}

}

### Test:

Josn Object which contains all the test need to be performed and configurations for the tests.

### absoluteDuplicateCheck:

This property will guide the framework to perform absolute duplicate check or not.

Value for this will be either true or false.

### absoluteDuplicateCheckFileName:

If absoluteDuplicateCheck is set to true, you have provide the location where it should write the records if any absolute duplicate records are found.

### duplicateCheck: -

This property will guide the framework to perform duplicate check or not.

Value for this will be either true or false.

### duplicateCheckColumns:

If duplicateCheck is set to true, you have to provide the unique key using which the framework will perform the group by operation for duplicate data identification.

### duplicateCheckFileName:

If duplicateCheck is set to true, you have provide the location where it should write the records if any duplicate data is found.

### nullCheck:-

This property will guide the framework to perform null check or not.

Value for this will be either true or false.

### input :

Json Object which contains configurations related to source of data

### isDirectHive:

This is to guide the framework that local hive needs to be used as source.

Value for this will be either true or false.

### isFile:

This is to guide the framework that the source of data is a csv file.

Value for this will be either true or false.

### filename:

If isFile is set to true, you have to provide the location of the file.

fileWithSchema:

If isFile is set to true, you have to provide is the file contains schema or not.

Value for this will be either true or false.

### driverName:

if isFile and isHiveDirect is set to false, you have to provide the JDBC details where t fetch the data for sanity check. JDBC driver class name should be defined in this property.

### databaseUrl:

JDBC URL should be defined in this property to connect to source database.

### databaseUserName:

Database username.

### databasePassword:

Database password.

### query:

If isFile is set to false, then this property should contain SQL query that will be executed to get the source data.

### Log:

Json Object contains information related to log file details.

### filename:

File name with path where log will be written into.